ИУ5-32Б,

Фролов Илья

Рубежный контроль 2

Вариант 22

Рубежный контроль представляет собой разработку тестов на языке Python.

1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

```
from operator import itemgetter


class ProgramLang:
    def __init__(self, id, name):
        self.id = id
        self.name = name


class Library:
    def __init__(self, id, name, doc_lib_href, prog_lang_id):
        self.id = id
        self.name = name
        self.doc_lib_href = doc_lib_href
        self.prog_lang_id = prog_lang_id


class ProgLangLib:
    def __init__(self, prog_lang_id, lib_id):
        self.prog_lang_id = prog_lang_id
        self.lib_id = lib_id


def get_one_to_many(prog_langs, libs):
    return [(lib.name, lib.doc_lib_href, pl.name)
            for pl in prog_langs
            for lib in libs
            if lib.prog_lang_id == pl.id]


def get_many_to_many(pl_libs, libs, prog_langs):
    many_to_many_temp = [(pl.name, ps.prog_lang_id, ps.lib_id)
                         for pl in prog_langs
                         for ps in pl_libs
                         if ps.prog_lang_id == pl.id]

    return [(lib.name, lib.doc_lib_href, pl_name)
            for pl_name, pl_id, lib_id in many_to_many_temp
            for lib in libs if lib.id == lib_id]


def first_task(lib_list):
    return sorted(lib_list, key=itemgetter(0))


def second_task(lib_list):
    res_2 = []
    temp_dict = dict()
```

```
    for i in lib_list:
        if i[2] in temp_dict:
            temp_dict[i[2]] += 1
        else:
            temp_dict[i[2]] = 1
    for i in temp_dict.keys():
        res_2.append((i, temp_dict[i]))

    res_2.sort(key=itemgetter(1), reverse=True)
    return res_2


def third_task(lib_list, end_ch):
    return [(i[0], i[2]) for i in lib_list if
str(i[0]).endswith(end_ch)]


def main():
    prog_langs = [
        ProgramLang(1, "C++"),
        ProgramLang(2, "Java"),
        ProgramLang(3, "Kotlin"),
    ]

    libs = [
        Library(1, "JUnit4", "https://kotlinlang.org/docs/jvm-test-
using-junit.html", 3),
        Library(2, "JUnit4", "https://junit.org/junit4/", 2),
        Library(3, "JUnit5", "https://junit.org/junit5/", 3),
        Library(4, "Cucumber",
"https://cucumber.io/docs/installation/java/", 3),
        Library(5, "iostream",
"https://en.cppreference.com/w/cpp/header/iostream", 1),
        Library(6, "stdlib",
"https://en.cppreference.com/w/cpp/header/cstdlib", 1)
    ]

    pl_libs = [
        ProgLangLib(1, 5),
        ProgLangLib(1, 6),
        ProgLangLib(3, 4),
        ProgLangLib(3, 3),
        ProgLangLib(2, 2),
        ProgLangLib(3, 1),
    ]

    one_to_many = get_one_to_many(prog_langs, libs)
    many_to_many = get_many_to_many(pl_libs, libs, prog_langs)

    print('Задание Б1')
    for lib in first_task(one_to_many):
        print(lib)

    print("\nЗадание Б2")
    print(second_task(one_to_many))

    print("\nЗадание Б3")
    print(third_task(many_to_many, '4'))


if __name__ == '__main__':
    main()
```

2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).
UNIT_TESTS.py

```
import main
from operator import itemgetter
import unittest


class TestMainMethods(unittest.TestCase):

    def setUp(self):
        self.prog_langs = [
            main.ProgramLang(1, "C++"),
            main.ProgramLang(2, "Java"),
            main.ProgramLang(3, "Kotlin"),
        ]

        self.libs = [
            main.Library(1, "JUnit4", "https://kotlinlang.org/docs/jvm-
test-using-junit.html", 3),
            main.Library(2, "JUnit4", "https://junit.org/junit4/", 2),
            main.Library(3, "JUnit5", "https://junit.org/junit5/", 3),
            main.Library(4, "Cucumber",
"https://cucumber.io/docs/installation/java/", 3),
            main.Library(5, "iostream",
"https://en.cppreference.com/w/cpp/header/iostream", 1),
            main.Library(6, "stdlib",
"https://en.cppreference.com/w/cpp/header/cstdlib", 1)
        ]

        self.pl_libs = [
            main.ProgLangLib(1, 5),
            main.ProgLangLib(1, 6),
            main.ProgLangLib(3, 4),
            main.ProgLangLib(3, 3),
            main.ProgLangLib(2, 2),
            main.ProgLangLib(3, 1),
        ]

        self.one_to_many = [
            ('JUnit5', 'https://junit.org/junit5/', 'Kotlin'),
            ('Cucumber', 'https://cucumber.io/docs/installation/java/',
'Kotlin'),
            ('JUnit4', 'https://junit.org/junit4/', 'Java'),
            ('JUnit4', 'https://kotlinlang.org/docs/jvm-test-using-
junit.html', 'Kotlin'),
            ('iostream',
'https://en.cppreference.com/w/cpp/header/iostream', 'C++'),
            ('stdlib',
'https://en.cppreference.com/w/cpp/header/cstdlib', 'C++')
        ]

    def test_first_task_method(self):
        result = main.first_task(self.one_to_many)
        reference = sorted(self.one_to_many, key=itemgetter(0))
        self.assertEqual(result, reference)

    def test_second_task_method(self):
        result = main.second_task(self.one_to_many)
        reference = [('Kotlin', 3), ('C++', 2), ('Java', 1)]
        self.assertEqual(result, reference)
```

```
    def test_third_task_method(self):
        many_to_many = [
            ('JUnit4', 'https://kotlinlang.org/docs/jvm-test-using-
junit.html', 'Kotlin'),
            ('JUnit4', 'https://junit.org/junit4/', 'Java'),
            ('JUnit5', 'https://junit.org/junit5/', 'Kotlin'),
            ('Cucumber', 'https://cucumber.io/docs/installation/java/',
'Kotlin'),
            ('iostream',
'https://en.cppreference.com/w/cpp/header/iostream', 'C++'),
            ('stdlib',
'https://en.cppreference.com/w/cpp/header/cstdlib', 'C++'),
        ]

        result = main.third_task(many_to_many, '4')
        reference = [('JUnit4', 'Java'), ('JUnit4', 'Kotlin')]
        self.assertEqual(sorted(result), sorted(reference))
         # Сравниваем отсортированные списки


if __name__ == '__main__':
    unittest.main()
```

Результаты тестирования:

```
A:\PycharmProjects\python_RK1\pythonProject1\.venv\Scripts\python.exe "A:/Program Files/PyCharm Community Edition 2024.2.1/plugins/pyth
Testing started at 20:44 ...
Launching unittests with arguments python -m unittest unit_tests.TestMainMethods in A:\PycharmProjects\python_RK1\pythonProject1\RK1


Ran 3 tests in 0.002s

OK

Process finished with exit code 0
```